◆用于<…>的函数

2024040731192_李畅_4.cpp

```
1  #include <stdio.h>
2  #include <cstdlib>       // 包含 system(),rand(),srand()函数的头文件
3  #include <ctime>         // 用于设置随机种子
4  #include <stdlib.h>      // 需要包含 malloc 的头文件
5  #include <iostream>      // c++库.是 IO 的库
6  #include <stdbool.h>     // bool
7
8  #define MaxSize 50       // 顺序表最大容限
9
10 using namespace std;     // 调用标准(std)命名空间
11
12 typedef int ElemType;
13
14 typedef struct
15 {
16     ElemType data[MaxSize];
17     int length;
18 } SqList;
19
20 void Menu();                                          // 控制面板
21
22 void InitList(SqList *&L);                            // 创建空表
23 void DestroyList(SqList *&L);                         // 销毁表
24
25 void CreateList(SqList *&L, ElemType a[], int n);     // 输入表
26 void DispList_1(SqList *L);                           // 输出表
27 void DispList_2(SqList *L);                           // 输出表
28
29 void Search_sys(SqList *L);                           // 查找系统
30 bool ListEmpty(SqList *&L);                           // 判断是否为空
31  int ListLength(SqList *L);                           // 返回表长度
32 bool GetElem(SqList *L, int i, ElemType &e);         // 按序查找
33  int LocateElem(SqList *L, int Bp, ElemType e);      // 按值查找
34
35 void upDate_sys(SqList *&L);                          // 修改系统
36 bool ListUpdate(SqList *&L, int i, ElemType e);      // 修改元素
37 bool ListInsert(SqList *&L, int i, ElemType e);      // 插入元素
38 bool ListDelete(SqList *&L, int i, ElemType &e);     // 删除数据
39
40 void main2(SqList *&L1, SqList *&L2);                 // 附加系统
41 void Sort(SqList *&L, int m);                         // 排序表
```

```
42  void ListUnion(SqList *&L1, SqList *&L2);        // 合并有序表
43
44   int input(char c);                               // sys 选择
45  bool Input(const char *s, int *n);                // 整型验证
46  void clearInputBuffer();                          // 清除缓冲区
47
48  int main()
49  {
50
51      int choice=0;
52      char c='N';
53      SqList *L;            // 表 1
54      InitList(L);
55
56      SqList *L1;           // 表 2
57      InitList(L1);
58
59      srand(time(NULL)); // 生成随机数种子
60
61      do
62      {
63          choice=0; c='N';
64          Menu();
65          cin>>(c);
66            choice=input(c);
67            clearInputBuffer();
68
69          switch(choice)
70          {
71              case 0: {             // ** 退出指令 **
72                  DestroyList(L);
73                  DestroyList(L1);
74                  cout<<"\n 表已销毁。\n";
75                     cout<<"\n 系统已退出。\n";
76                     break;
77              }
78
79              case 1: {             // ** 建立实验表 1 **
80                  int n=0,i;
81
82                  if(!Input("\n 请输入表 1 的长度:",&n)) break;
83
84                  n<0?n=0:n;                        // 限制长度
85                  n>MaxSize?n=MaxSize:n=n;
```

```
86
87                  ElemType *a = new ElemType[n];          // 动态分配数组
88
89                  for(i=0;i<n;i++)
90                      a[i]=rand(); //生成试验数据
91
92                  CreateList(L,a,n);
93
94                  cout<<"\n---- 顺序表 1 已建立! ----";
95                  delete[] a;       // 释放数组
96                  break;
97              }
98
99              case 2: {             // ** 查找系统 **
100                 Search_sys(L);
101                 break;
102             }
103
104             case 3: {             // ** 修改系统 **
105                 upDate_sys(L);
106                 break;
107             }
108
109             case 4: {             // ** 附加实验 **
110                 main2(L,L1);
111                 break;
112             }
113             default: cout<<"\n---- 输入错误! ----\n";
114         }
115     ////////////////////////////////////
116         cout<<"\n\n 按任意键继续....";
117         c=getchar();
118     } while(choice);
119
120     return 0;
121 }
122
123 ////////////////////////////////////////////////////////////////////////////////////////////////////
124 // ** 控制面板 **
125
126 void Menu()
127 {
128     system("cls");
129     cout<<"******整形顺序表操作******\n";
```

```
130     cout<<"-------------------------\n";
131     cout<<"         1   建立实验表 1         \n";
132     cout<<"         2    查询系统            \n";
133     cout<<"         3    修改系统            \n";
134     cout<<"-------------------------\n";
135     cout<<"         4    附加功能            \n";
136     cout<<"-------------------------\n";
137     cout<<"         0    退出程序            \n";
138     cout<<"-------------------------\n";
139     cout<<"***************************\n";
140     cout<<"请选择： ";
141 }
142
143 ////////////////////////////////////////////////////////////////////////////////////////////////////
144 // ** 创建空表 **
145
146 void InitList(SqList *&L)
147 {
148     L=(SqList *)malloc(sizeof(SqList));
149     L->length=0;
150 }
151
152 ////////////////////////////////////////////////////////////////////////////////////////////////////
153 // ** 销毁表 **
154
155 void DestroyList(SqList *&L)
156 {
157     free(L);
158 }
159
160 ////////////////////////////////////////////////////////////////////////////////////////////////////
161 // ** 判断是否为空 **      ps:实际改为三值表示会更好
162
163 bool ListEmpty(SqList *&L)
164 {
165     return(L->length==0);
166 }
167
168 ////////////////////////////////////////////////////////////////////////////////////////////////////
169 // ** 返回表长度 **
170
171 int ListLength(SqList *L)
172 {
173     return(L->length);
```

```
174 }
175
176 ////////////////////////////////////////////////////////////////////////////////////////////////////
177 // ** 输入表 **
178
179 void CreateList(SqList *&L, ElemType a[], int n)
180 {
181     int i=0,k=0;
182     while(i<n)
183     {
184         L->data[k]=a[i];
185         k++;i++;
186     }
187     L->length=k;
188 }
189
190 ////////////////////////////////////////////////////////////////////////////////////////////////////
191 // ** 输出表 **
192
193 void DispList_1(SqList *L)
194 {
195     printf("\n- - - - - - - - - - - - - - - -\n\n");
196     for(int i=0;i<L->length;i++) {
197         printf(" %d\t",L->data[i]);
198         if((i+1)%4==0) printf("\n");
199     }
200     printf("\n_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _\n");
201 }
202
203 void DispList_2(SqList *L)
204 {   // 显示所有元素与其序号的对应关系
205     int i;
206     ElemType e;
207     cout<<"\n 当前表中的元素及其序号对映：\n";
208     cout<<"_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _\n";
209     for(i=1; i<=ListLength(L); i++)
210         if(GetElem(L, i, e))
211             cout<<"  序号 "<<i<<": \t\t"<<e<<endl;
212     cout<<"_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _\n";
213 }
214
215 ////////////////////////////////////////////////////////////////////////////////////////////////////
216 // ** 查找系统 **
217
```

```
218 void Search_sys(SqList *L)
219 {
220     int choice;
221     char c='N';
222      do {
223         choice=999; c='N';
224         system("cls");
225         cout<<"******整形顺序表操作******\n";
226         cout<<"------------------------\n";
227         cout<<"---------查询系统---------\n";
228         cout<<"        1   返回表长度          \n";
229         cout<<"        2   遍历显示表          \n";
230         cout<<"        3 按序查找元素          \n";
231         cout<<"        4 按值查找元素          \n";
232         cout<<"        0    退出程序           \n";
233         cout<<"------------------------\n";
234         cout<<"**************************\n";
235          printf("\n 请选择 (0-4): ");
236          cin>>(c);
237          choice=input(c);
238          clearInputBuffer();
239
240          switch(choice) {
241            case 0: return;// 退出子程序
242
243            case 1: {
244                 printf("\n---- 顺序表长度为%d ----",ListLength(L));
245                 break;
246            }
247
248            case 2: {
249                 DispList_1(L);
250                 break;
251            }
252
253              case 3: {
254                int i; bool bl;
255                ElemType e;
256
257                if(!Input("\n 请输入要查找的序号:",&i)) break;
258
259                bl=GetElem(L, i, e);
260                printf("\n----- %s 该序号 -------\n",bl?"查到":"未查到");
261                //bl?printf("值为:%d",e):0;
```

```
262                     if(bl)     printf("值为:%d",e);
263                     break;
264                 }
265
266                   case 4: {
267                     int Bp=0,count=0;
268                     ElemType e;
269
270                     if(!Input("\n 请输入要查找的元素值:",&e)) break;
271
272                     cout<<"\n\n------------------------\n\n";
273                     do {
274                             Bp=LocateElem(L, Bp, e);
275                             count++;
276                             if(Bp!=0) printf("第%d 个匹配元素的位置：%d\n", count, Bp);
277                     } while(Bp);
278                     if(count==1) printf("未找到值为 %d 的元素位置!\n",e);
279                     cout<<"\n------------------------\n";
280                     break;
281                 }
282                   default: cout<<"\n---- 输入错误! ----\n";
283             }
284     ////////////////////////////////////
285           cout<<"\n\n 按任意键继续....";
286           c=getchar();
287       } while (choice!=0);
288 }
289
290 // ** 按序查找 **
291
292 bool GetElem(SqList *L, int i, ElemType &e)
293 {
294     if(i<1 || i>L->length)
295         return false;
296     e=L->data[i-1];
297     return true;
298 }
299
300 // ** 按值查找 **
301
302 int LocateElem(SqList *L, int Bp, ElemType e)
303 {                   // Breakpoint 查询断点
304     int i;
```

```
305     for(i=Bp;i<L->length;i++)
306         if(L->data[i]==e)
307             return  i+1;
308     return  0;
309 }
310
311 ////////////////////////////////////////////////////////////////////////////////////////////////////
312 // **  修改系统  **
313
314 void  upDate_sys(SqList  *&L)
315 {
316     int  choice;
317     char  c='N';
318      do  {
319         choice=999;  c='N';
320         system("cls");
321         cout<<"******整形顺序表操作******\n";
322         cout<<"------------------------\n";
323           cout<<"---------修改系统---------\n";
324           cout<<"          1  定点插入元素          \n";
325           cout<<"          2  定点修改元素          \n";
326           cout<<"          3  查值修改元素          \n";
327         cout<<"          4  定点删除元素          \n";
328         cout<<"          5  查值删除元素          \n";
329           cout<<"          0     退出程序           \n";
330           cout<<"------------------------\n";
331         cout<<"**************************\n";
332           printf("\n 请选择  (0-5):  ");
333           cin>>(c);
334           choice=input(c);
335           clearInputBuffer();
336
337           switch(choice)  {
338             case  0:  return;//  退出子程序
339
340             case  1:  {
341                 int  i;
342                 ElemType  e;
343
344                 if(ListEmpty(L)==false)
345                 {
346                     DispList_2(L);
347                 }  else  {
348                     printf("\n 表为空,请注意只能在首位插入!!!\n");
```

```
349                 }
350
351                 printf("\n 请输入插入点 (1-%d) :",L->length==MaxSize?50:L->length+1);
352
353                 if(!Input(" ",&i)) break;
354
355                 if(!Input("\n 请输入元素:",&e)) break;
356
357                 printf("\n---- 插入操作%s ----\n",ListInsert(L,i,e)?"成功！":"失败");
358                 printf("\n 此时顺序表长度为 %d / %d ",ListLength(L),MaxSize);
359                 break;
360             }
361
362             case 2: {
363                 int i;
364                 ElemType e;
365
366                 if(ListEmpty(L)==false)
367                 {
368                     DispList_2(L);
369                 } else {
370                     printf("\n 表为空,修改操作已退出!\n");
371                     break;
372                 }
373
374                 printf("\n 请输入修改点 (1-%d) :",L->length);
375
376                 if(!Input("",&i)) break;
377
378                 if(!Input("\n 请输入元素:",&e)) break;
379
380                 printf("\n---- 修改操作%s ----\n",ListUpdate(L,i,e)?"成功！ ":"失败");
381                 printf("\n 此时顺序表长度为 %d / %d ",ListLength(L),MaxSize);
382                 break;
383             }
384
385               case 3: {
386                 int Bp=0,count=0;
387                 char c='N';
```

```
388                 ElemType  e,e1;
389
390                 if(!Input("\n 请输入要查找的元素值:",&e))  break;
391
392                 cout<<"\n\n-----------------------\n\n";
393                 while(1)
394                 {
395                         c='N';  e1=e;
396                         Bp=LocateElem(L,  Bp,  e);
397
398                         if(Bp!=0)  {
399                             count++;
400                             printf("第%d 个匹配元素的位置：%d\n",  count,  Bp);
401                         }  else  {
402                             cout<<"\n--------  表已查完  --------\n";
403                             break;
404                         }
405
406                         cout<<"\n 是否修改该处的值？（ Y  or  N ）:";
407                         cin>>(c);  clearInputBuffer();
408
409                         if(c=='y'  ||  c=='Y')
410                         {
411                             if(!Input("\n 请输入元素:",&e1))  break;
412                             printf("\n  ----  修改操作%s  ----\n",ListUpdate(L,Bp,e1)?"成功！ ":"失败");
413                         }
414
415                         cout<<"\n 是否继续查找该值？（ Y  or  N ）:";
416                         cin>>(c);  clearInputBuffer();
417
418                         if(c=='n'  ||  c=='N')  break;
419                         cout<<"\n-----------------------\n\n";
420                 }
421                 if(count==0)
422                     printf("  未找到值为  %d  的元素位置!\n",e);
423                 else
424                     printf("\n     共找到  %d  个匹配元素.\n",count);
425                 cout<<"\n-----------------------\n";
426                 break;
427             }
428
429               case  4:  {
```

```
430                 int i; c='N';
431                 bool isOK=0;
432                 ElemType e;
433
434                 if(ListEmpty(L)==true)
435                 {
436                     printf("\n 当前表为空,删除操作已退出!");
437                     break;
438                 }
439
440                 DispList_2(L);
441
442                 printf("\n 请输入删除点: (1-%d) :",L->length==MaxSize?50:L->length+1);
443                 if(!Input(" ",&i)) break;
444
445                 cout<<"\n 要删除的元素为:";
446                 printf("%d\n",GetElem(L,i,e)?e:0);
447
448                 cout<<"\n 是否确认删除? ( Y or N ):";
449                 cin>>(c); clearInputBuffer();
450
451                 if(c=='Y' || c=='y')
452                 {
453                     isOK=ListDelete(L,i,e);
454                     printf("\n---- 删除操作%s ----\n",isOK?"成功! ":"失败!!!");
455                     if(isOK) printf("\n 顺序表长度为 %d/%d \n",ListLength(L),MaxSize);
456
457                 } else {
458                     cout<<"\n---- 删除操作已取消! ----";
459                 }
460                 break;
461             }
462
463             case 5: {
464                 int Bp=0,count=0;
465                 char c='N';
466                 bool isOK=0;
467                 ElemType e,e1;
468
469                 if(!Input("\n 请输入要查找的元素值:",&e)) break;
470
```

```
471                 cout<<"\n\n------------------------\n\n";
472                 while(1)
473                 {
474                         c='N'; isOK=0;
475                         Bp=LocateElem(L,Bp,e);
476
477                         if(Bp!=0) {
478                             count++;
479                             printf("第%d 个匹配元素的位置：%d\n", count, Bp);
480                         } else {
481                             cout<<"\n-------- 表已查完 --------\n";
482                             break;
483                         }
484
485                         cout<<"\n 是否删除该处的值？( Y or N ):";
486                         cin>>(c); clearInputBuffer();
487
488                         if(c=='y' || c=='Y')
489                         {
490                             isOK=ListDelete(L,Bp,e);
491                             printf("\n ---- 删除操作%s ---- \n",isOK?"成功！":"失败!!!");
492                             if(isOK) printf("\n 顺序表长度为 %d/%d \n",ListLength(L),MaxSize);
493                         }
494
495                         cout<<"\n 是否继续查找该值？( Y or N ):";
496                         cin>>(c); clearInputBuffer();
497
498                         if(c=='n' || c=='N') break;
499                         cout<<"\n------------------------\n\n";
500                 }
501                 if(count==0)
502                     printf(" 未找到值为 %d 的元素位置!\n",e);
503                 else
504                     printf("\n    共找到 %d 个匹配元素.\n",count);
505                 cout<<"\n------------------------\n";
506                 break;
507             }
508               default: cout<<"\n---- 输入错误！----\n";
509         }
510     ////////////////////////////////////
511         cout<<"\n\n 按任意键继续....";
```

```
512             c=getchar();
513         } while (choice!=0);
514 }
515
516 // ** 修改元素 **
517
518 bool ListUpdate(SqList *&L, int i, ElemType e)
519 {
520     if(i<1 || i>L->length || i>MaxSize)
521         return false;
522     i--;
523     L->data[i]=e;
524     return true;
525 }
526
527 // ** 插入元素 **
528
529 bool ListInsert(SqList *&L, int i, ElemType e)
530 {
531     int j;
532     if(i<1 || i>L->length+1 || L->length==MaxSize)
533         return false;
534     i--;
535     for(j=L->length;j>i;j--)
536         L->data[j]=L->data[j-1];
537     L->data[i]=e;
538     L->length++;
539     return true;
540 }
541
542 // ** 删除数据 **
543
544 bool ListDelete(SqList *&L, int i, ElemType &e)
545 {
546     int j;
547     if(i<1 || i>L->length)
548         return false;
549     i--;
550     e=L->data[i];
551     for(j=i;j<L->length-1;j++)
552         L->data[j]=L->data[j+1];
553     L->length--;
554     return true;
555 }
```

```
556
557 ////////////////////////////////////////////////////////////////////////////////////////////////////
558 // ** 选做实验 **
559
560 void main2(SqList *&L1, SqList *&L2)
561 {
562     int choice;
563     char c='N';
564      do {
565         choice=999; c='N';
566         system("cls");
567         cout<<"******整形顺序表操作******\n";
568         cout<<"------------------------\n";
569          cout<<"----------附加实验--------\n";
570         cout<<"          1   建立实验表 2          \n";
571         cout<<"          2   有序表的并            \n";
572          cout<<"          0    退出程序             \n";
573          cout<<"------------------------\n";
574         cout<<"**************************\n";
575          printf("\n 请选择 (0-2): ");
576          cin>>(c);
577          choice=input(c);
578          clearInputBuffer();
579
580          switch(choice) {
581            case 0: return;// 退出子程序
582
583            case 1: {
584                int n=0,i;
585
586                if(!Input("\n 请输入表 2 的长度:",&n)) break;
587
588                n<0?n=0:n;                          // 限制长度
589                n>MaxSize?n=MaxSize:n=n;
590
591                ElemType *a = new ElemType[n];          // 动态分配数组
592
593                for(i=0;i<n;i++)
594                    a[i]=rand(); //生成试验数据
595
596                CreateList(L2,a,n);
597
598                cout<<"\n---- 顺序表 2 已建立! ----";
599                delete[] a; // 释放数组
```

```
600                 break;
601             }
602
603             case 2: {
604                 int overflow=0;      // 判断是否溢出
605                 if(ListEmpty(L2)!=0)
606                 {
607                     cout<<"\n---- 表 2 为空,程序已退出! ----";
608                     break;
609                 }
610
611                 if(ListEmpty(L1) || L1->length==MaxSize)
612                 {
613                     cout<<"\n ---- 表 1 为空或已满! ----";
614                     break;
615                  }
616
617                 overflow=L1->length+L2->length;
618
619                 Sort(L1,L1->length);
620                 Sort(L2,L2->length);
621
622                 cout<<"\n\n   ------- 两表已排序! ------";
623                 cout<<"\n 表 1 排序如下:\n";
624                 DispList_1(L1);
625                 cout<<"\n 表 2 排序如下:\n";
626                 DispList_1(L2);
627
628                 ListUnion(L1,L2);   // 调用合并函数
629
630                 cout<<"\n\n   ------- 两表已合并! ------";
631                 cout<<"\n 表 1 结果如下:\n";
632                 DispList_1(L1);
633                 printf("\n 顺序表 1 长度为 %d/%d \n", ListLength(L1), MaxSize);
634                 printf("\n   -------- 运算%s --------", overflow>50?"溢出":"未溢出");
635
636                 InitList(L2);     // 重建表 2 以便继续运行程序
637                 break;
638             }
639             default: cout<<"\n      ---- 输入错误! ----\n";
640         }
641     ////////////////////////////////////
```

```
642             cout<<"\n\n 按任意键继续....";
643             c=getchar();
644         } while (choice!=0);
645 }
646
647 // ** 线性表的排序 **
648
649 void Sort(SqList *&L, int m)
650 {
651     int i,j;
652     int ts;
653
654     for(i=1;i<=m;i++)                       // 冒泡排序
655         for(j=0;j<m-i;j++)
656             if(L->data[j]>L->data[j+1])
657             {
658                 ts=L->data[j];
659                 L->data[j]=L->data[j+1];
660                 L->data[j+1]=ts;
661             };
662 }
663
664 // ** 有序表的并 **
665
666 void ListUnion(SqList *&L1, SqList *&L2)
667 {
668     int i=0,j=0;
669     while(i<L1->length && j<L2->length)
670     {
671         if(L1->data[i] < L2->data[j])
672             i++;                            // 若 L1 小，则跳过
673         else
674         {
675             ListInsert(L1,i+1,L2->data[j]);  // 反之，将 L2 插入
676             j++;
677         }
678     }
679     if(i==L1->length)                       // 当 L1 到表尾时
680         while(j<L2->length)                 // 将 L2 接到 L1 后
681         {
682             ListInsert(L1,i+1,L2->data[j]);
683             i++;j++;
684         }
685     DestroyList(L2);
```

```
686 }
687
688 ////////////////////////////////////////////////////////////////////////////////////////////////////
689 // ** sys 选择验证 **
690
691 int input(char c)
692 {
693     if('0'<=c && '9'>=c)
694         return (int )c-48;
695     else
696         return -1;
697 }
698
699 ////////////////////////////////////////////////////////////////////////////////////////////////////
700 // ** 整型输入验证 **
701
702 bool Input(const char *s, int *n)
703 {
704     if(s) printf("%s",s);
705      if(scanf("%d",n)!=1)
706     {
707           printf("\n 输入字符非法!");
708           clearInputBuffer();
709           return 0;
710      }
711      clearInputBuffer();
712      return 1;
713 }
714
715 ////////////////////////////////////////////////////////////////////////////////////////////////////
716 // ** 清除缓冲区 **
717
718 void clearInputBuffer() {
719      while(getchar()!='\n');
720 }
721
```